NODE CONGRESS

Five ways of taking advantage of Verdaccio, your private and proxy Node.js registry

Juan Picado

18th February 2022

Juan Picado

Senior Front-End Engineer at mobile.de (Adevinta)

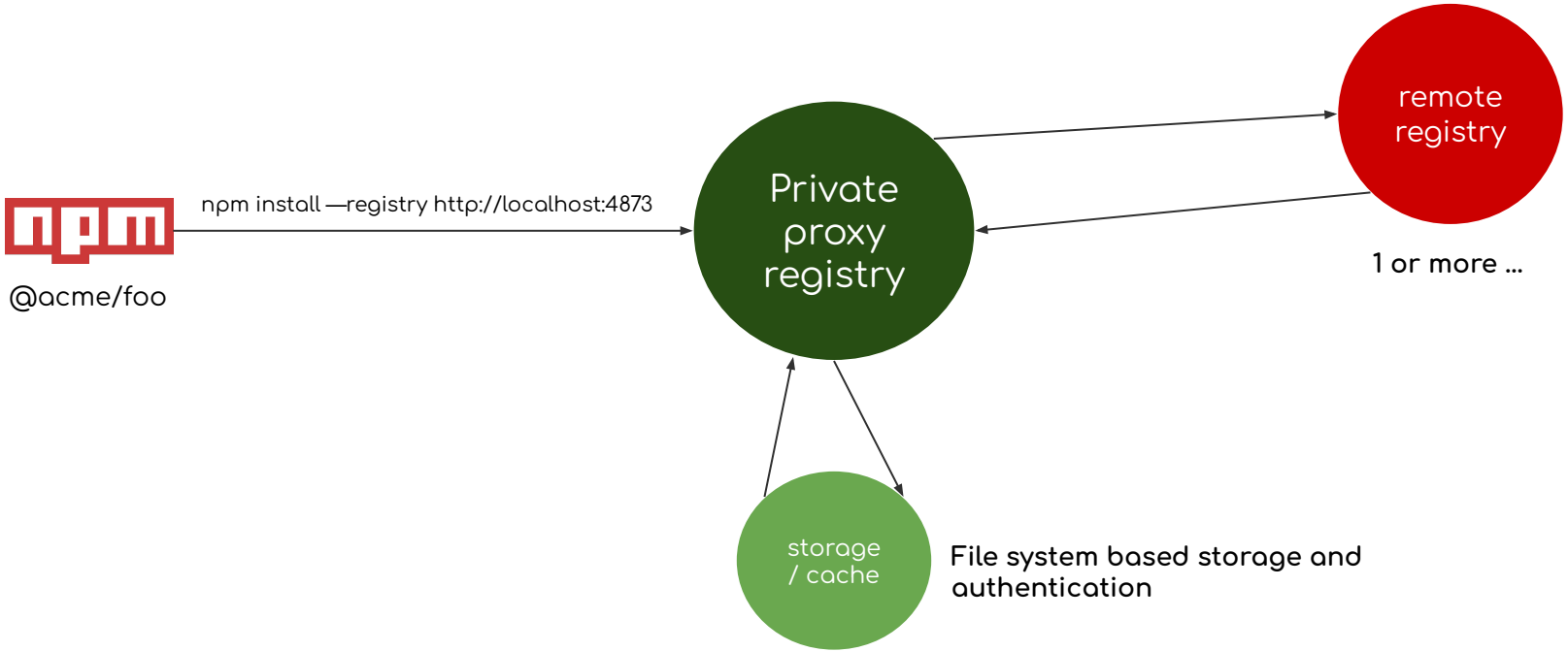
Open Source Maintainer (Verdaccio)

Berlin, Germany

@jotadeveloper

juanpicado

Verdaccio is a lightweight private proxy registry for Node.js built in JavaScript



Installing Verdaccio

Running Verdaccio in a terminal

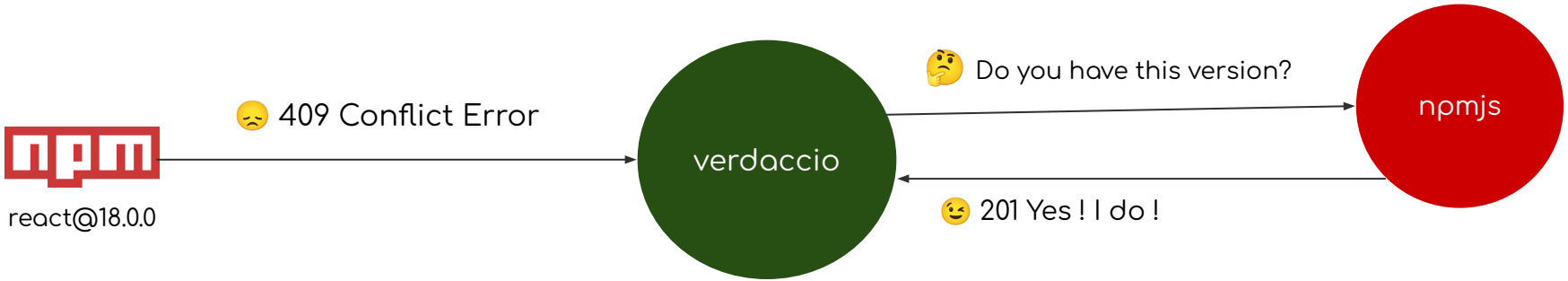
1 Personal Development

Publishing private packages

Demo, let's publish a private package with npm workspaces ...

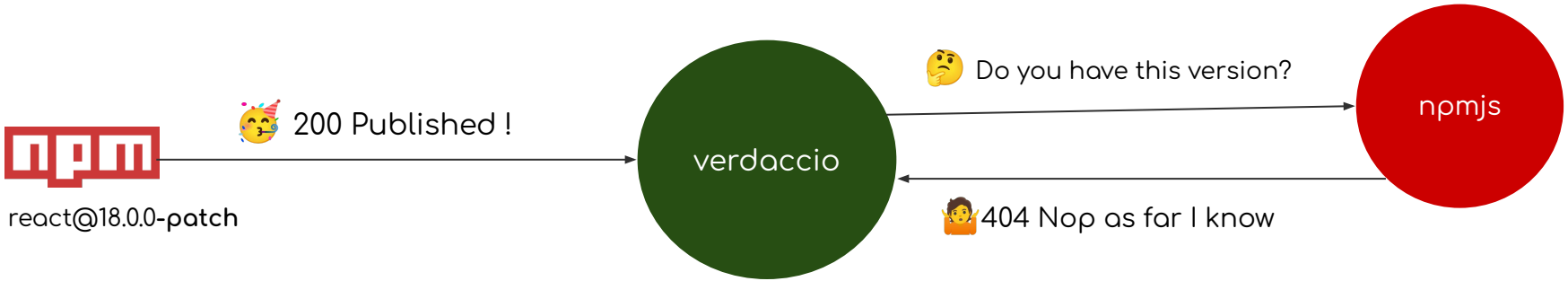
Constraints, cannot override remote published versions ...

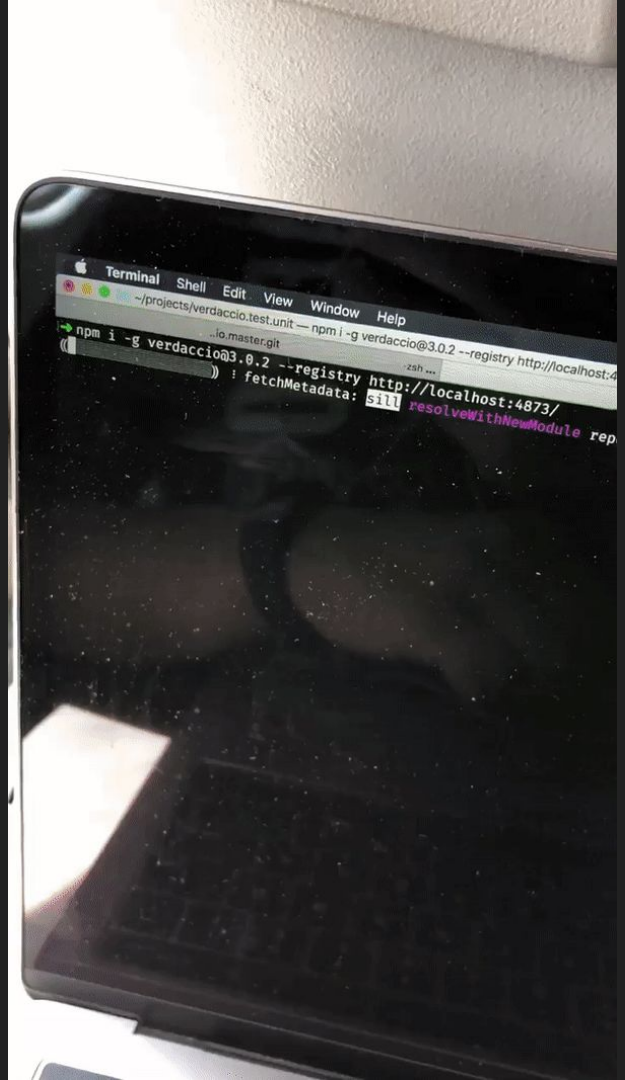
npm publish —registry http://localhost:4873



Constraints, cannot override remote published versions ...

npm publish —registry http://localhost:4873





Installing in offline mode ...

https://twitter.com/jotadeveloper/status/1015333131002564608

```yaml
# config.yaml

storage: /home/user/.local/share/verdaccio/storage

# Publishing offline
publish_offline: true

packages:
  '@acme/*':
    access: $authenticated
    publish: $all
    ## ofline
    # proxy: npmjs
  '@*/*':
    access: $all
    publish: $all
    unpublish: $authenticated
    proxy: npmjs
```

Publishing with no network using publish_offline

2 Project productivity

Continuous Integration and private registries

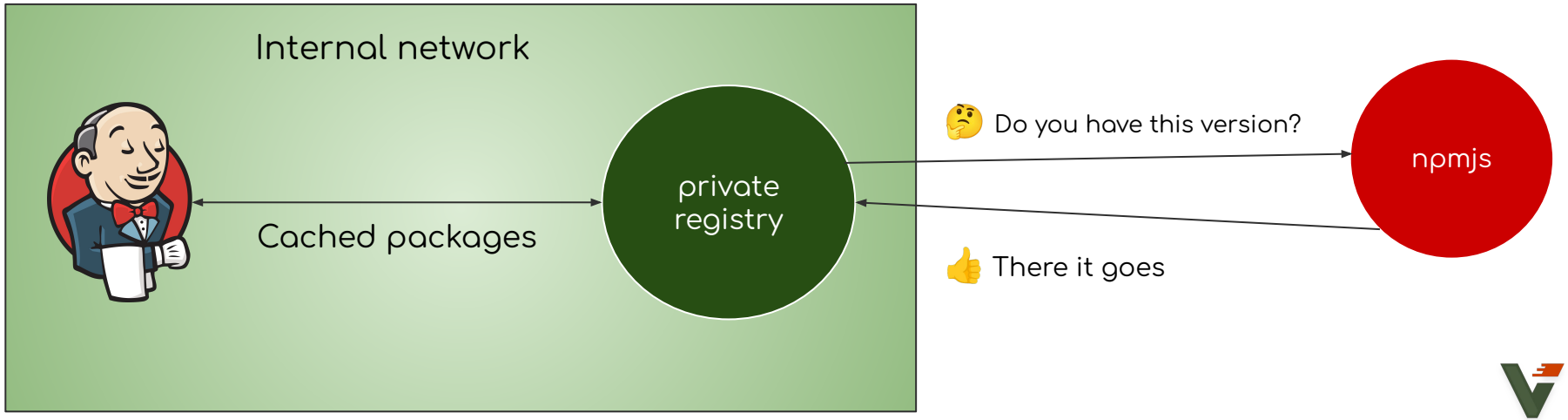
Matteo Collina
@matteocollina

I'm still surprised that having a private npm registry is not a standard across dev teams. Private npm, npm enterprise, artifactory, verdaccio: use what you prefer, but if you are not using them you are missing a lot.

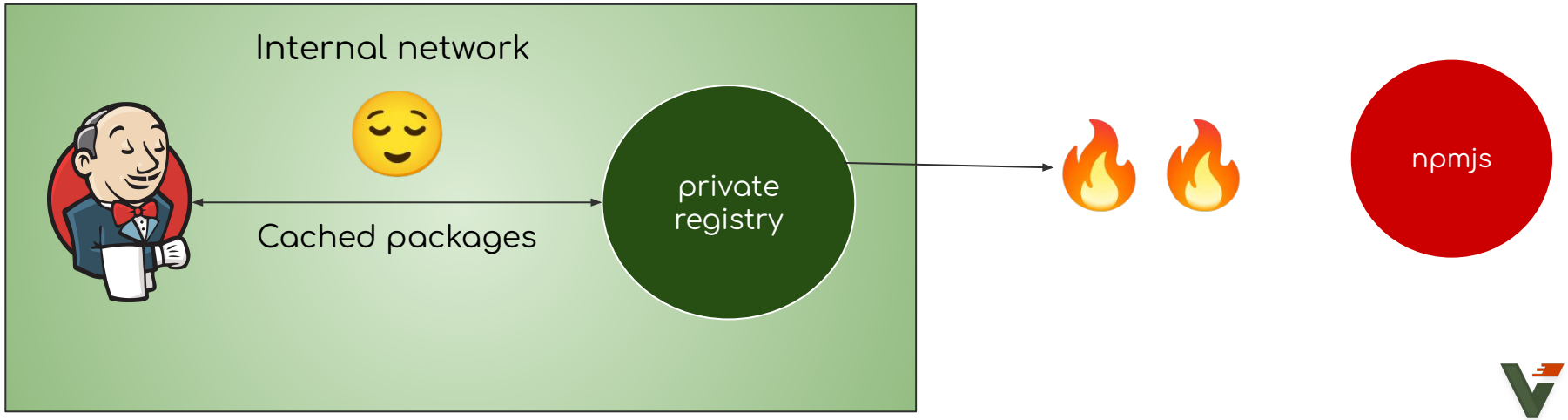
Traducir Tweet

6:49 p. m. · 26 feb. 2019 · Twitter Web Client

Private registries increase reliability in your builds and development workflows



Private registries increase reliability in your builds and development workflows



Private registries increase reliability in your builds and development workflows

```
uplinks:
  public:
    url: https://registry.npmjs.org
    maxage: 2w # 2 weeks
    timeout: 30s  # 30 seconds
    cache: false # only cache manifest
    strict_ssl: true
    max_fails: 10 # maximum failures
```

3 Improve Security

Protecting your packages

https://blog.includesecurity.com/2021/02/dependency-confusion-when-are-your-npm-packages-vulnerable/

```
uplinks:
  npmjs:
    url: https://registry.npmjs.org/
packages:
  '@acme/*':
    access: $authenticated
    publish: $authenticated
    unpublish: $authenticated
    # proxy: npmjs
```

Remove the proxy property for private packages this is highly recommendable.

```
{
  "name": "@acme/auth",
  "private": "true",
  "version": "1.0.0",
  "description": "always use scoped packages in your organization",
  "dependencies": {
    "@acme/internal": "4.x",
    "external": "1.x"
  }
}
```

Protect your projects in the client side
https://snyk.io/blog/ten-npm-security-best-practices/

Enable rate limiting for critical endpoints

```
Request URL: https://xxx.your.domain/-/verdaccio/data/packages
access-control-allow-origin: *
content-security-policy: connect-src 'self'
date: Sat, 29 Jan 2022 14:06:29 GMT
etag: "ba022b843888513a6218d751c275ab62"
server: nginx/1.14.0 (Ubuntu)
x-content-type-options: nosniff
x-frame-options: deny
x-ratelimit-limit: 1000
x-ratelimit-remaining: 998
x-ratelimit-reset: 1643465244
x-robots-tag: noindex, nofollow, nosnippet, noarchive
x-xss-protection: 1; mode=block
```

```
userRateLimit:
  windowMs: 50000
  max: 1000
```

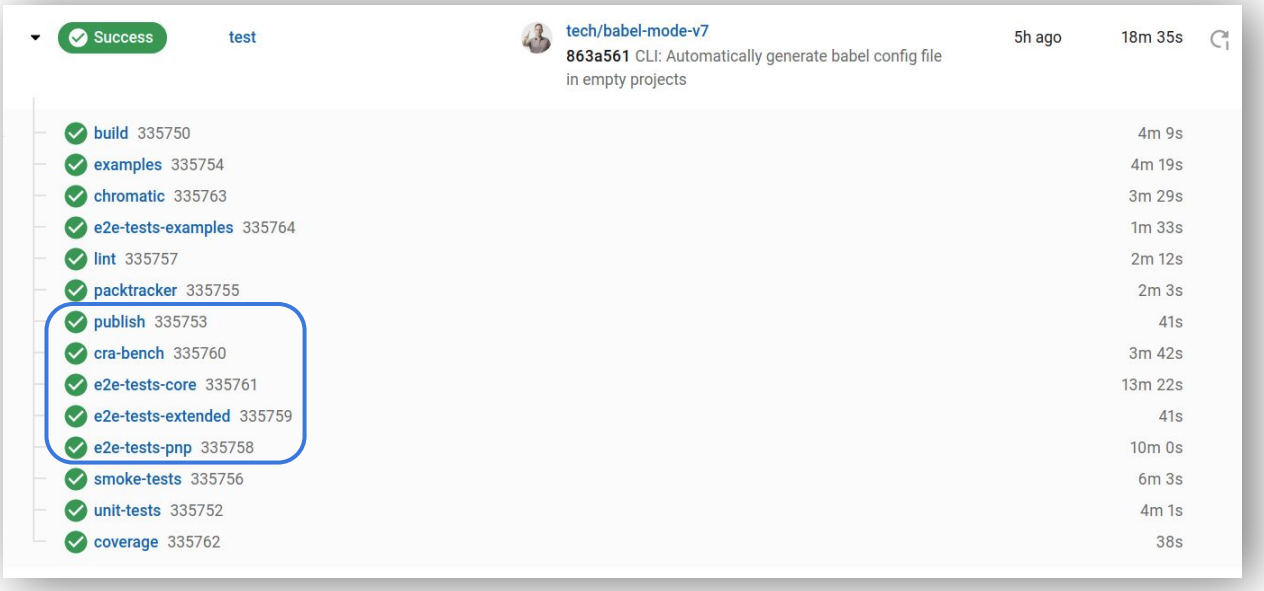
4 End to End Testing

Test the integrity of your packages

Publishing a package to thousands of users that download it every week requires high integrity, End to End your packages publishing in a registry ensure quality.

Storybook

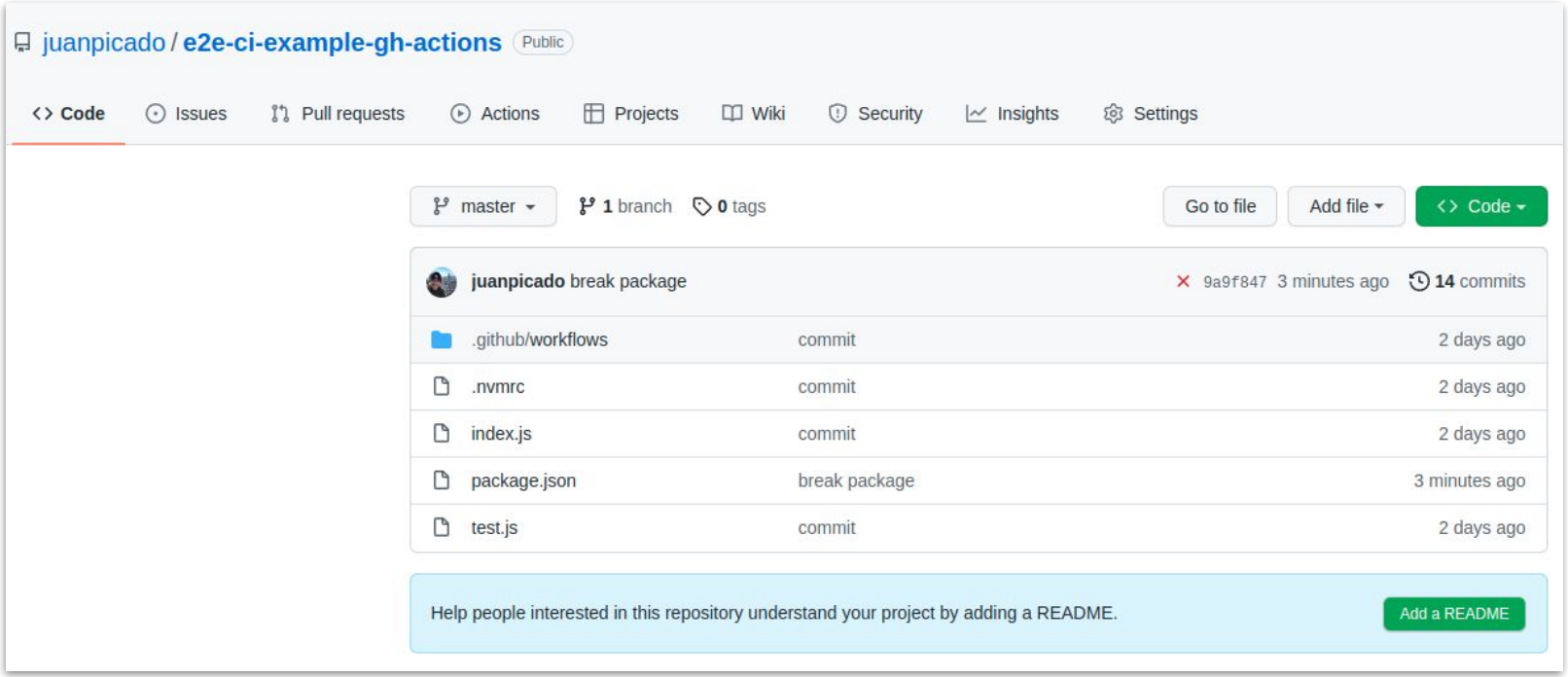
Test publishing your packages on every Pull Request



Demo, run a End to End with **GitHub Actions** and Docker ...

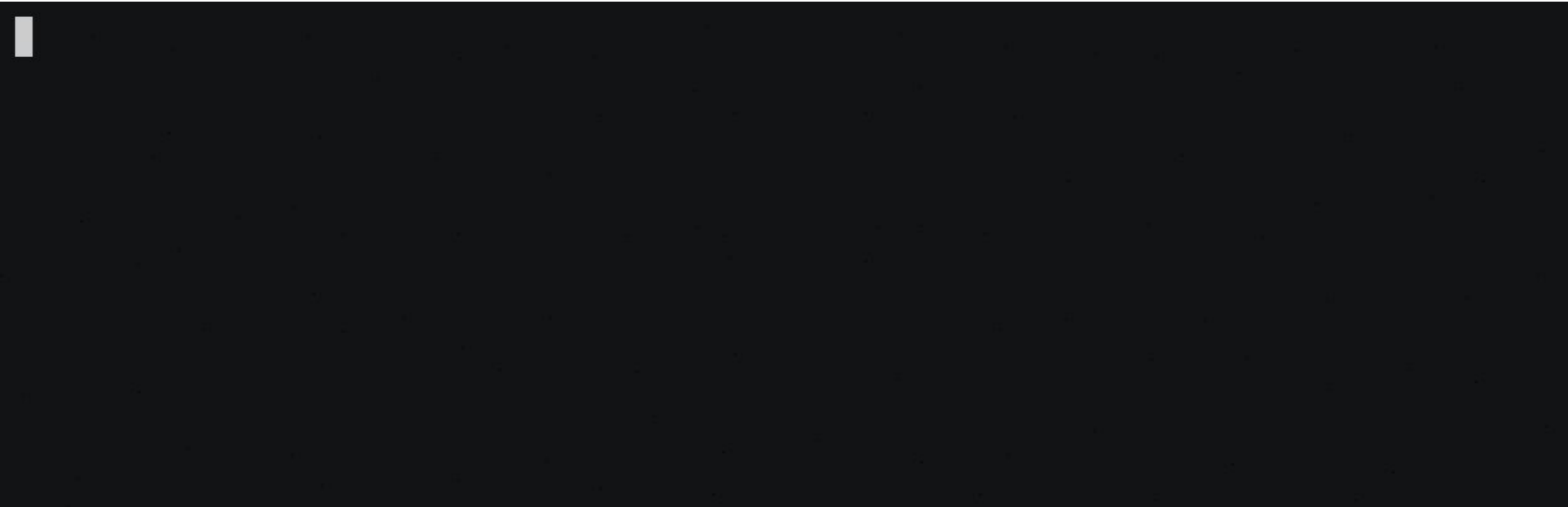
https://github.com/juanpicado/e2e-ci-example-gh-actions

Demo, run a End to End with **GitHub Actions** and Docker ...



Use memory plugins to speed up test

```
store:
  memory:
    limit: 1000
```



Dive into open source project for more learnings, every project is different.

```
# Babel.js
function startLocalRegistry {
  # Start local registry
  tmp_registry_log=`mktemp`
  echo "Registry output file: $tmp_registry_log"
  (cd && nohup npx verdaccio@~5.1.2 -c $1 &>$tmp_registry_log &)
  YARN_IGNORE_PATH=1 yarn global add verdaccio-memory@~10.0.0
  # Wait for Verdaccio to boot
  grep -q "http address" <(tail -f $tmp_registry_log)
  ...
```

With **bash** scripts

Dive into open source project for more learnings, every project is different.

```ts
// Angular-CLI
import { ChildProcess, spawn } from 'child_process';

export function createNpmRegistry(withAuthentication = false): ChildProcess {
  // Setup local package registry
  const registryPath = mkdtempSync(join(realpathSync(tmpdir()), 'angular-cli-e2e-registry-'));

  copyFileSync(
    join(__dirname, '../../', withAuthentication ? 'verdaccio_auth.yaml' : 'verdaccio.yaml'),
    join(registryPath, 'verdaccio.yaml'),
  );

  return spawn('node', [require.resolve('verdaccio/bin/verdaccio'), '-c', './verdaccio.yaml'], {
    cwd: registryPath,
    stdio: 'inherit',
  });
```

With **spawn** from **child_process** module

Dive into open source project for more learnings, every project is different.

```
// StoryBook
import startVerdaccioServer from 'verdaccio';

startVerdaccioServer(config, 6000, cache, '1.0.0', 'verdaccio', function onReady() {
  // do something
});
```

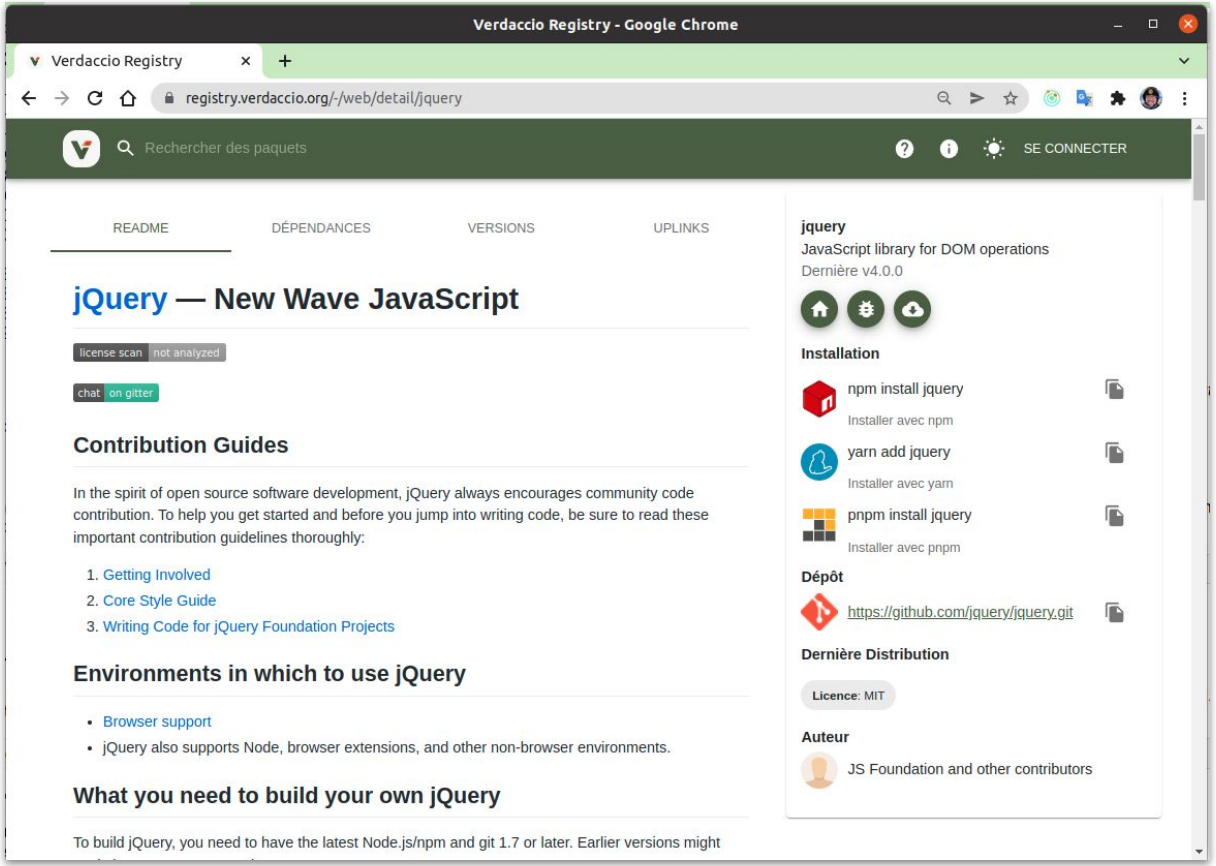
Programmatically with the verdaccio module

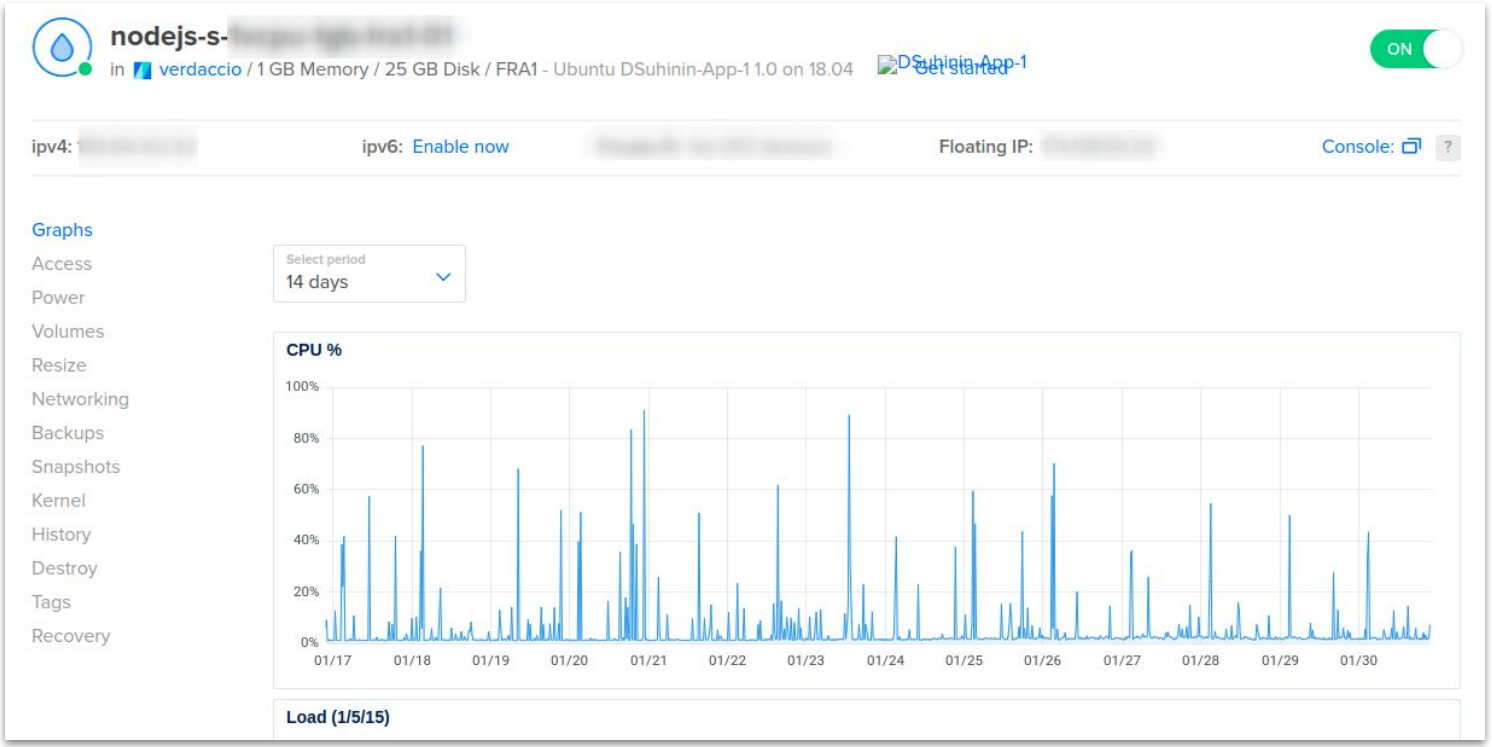
Storybook

5 Hosting a Registry

Security and flexibility

A very easy
deployment





pm2 + Nginx + Verdaccio

https://verdaccio.org/blog/2018/11/19/setting-up-verdaccio-on-digitalocean/

Extend the default configuration

- Use a different storage (S3, Minio)
- Use different authentication
  - Eg: GitLab, GitHub Oauth, etc ...
- Use your own User Interface
- Apply your own express middleware
  - Eg: npm audit (middleware plugin)

Thank you

https://verdaccio.org/

Juan Picado

@jotadeveloper

juanpicado